Advances in Computational Sciences and Technology
ISSN 0973-6107 Volume 10, Number 2 (2017) pp. 255-265

http://www.ripublication.com

Effect of Network Interconnects in Parallel Computing Systems on Parallelized Bioinformatics Applications

***Dhruv Chander Pant**
Ph.D Research Scholar,
I.K.G. PTU, Kapurthala, (Punjab) India.

****OP Gupta**

Associate Professor, School of Electrical Engg. and I.T.,
PAU, Ludhiana, (Punjab) India.

Abstract

Over the past few years due to various advancements in the field of molecular biology, an explosive growth in molecular databases has occurred. Along with management of huge molecular databases a heavy amount of computations is also required in various bioinformatics applications e.g. sequence analysis, phylogenetic analysis, gene expression analysis etc which is very time consuming. To full fill the need of performance in these applications, various bioinformatics algorithms have been parallelized to run on different architectures. One of the important architecture is the parallel computing systems. Considerable speedups are being obtained in the parallelized versions of different bioinformatics applications.

In this paper, various parallelized implementations of bioinformatics algorithms are studied. Along with, various parallel architectures used for algorithms and available technologies are also discussed. It is being analyzed that in all the parallel implementations of bioinformatics applications, the speed of network interconnects and communication media were always limiting factors to accelerate the performance. So high speed communication media and network interconnects need to be used with parallel computing systems, especially for bioinformatics applications. As in molecular biology-

applications and tools handle query sequences of varying size, so quantitative analysis of execution time for jobs of varying size is done. Jobs of varying size are executed on a cluster of 10 compute nodes with Basic Local Alignment Search Tool (BLAST). The cluster was privileged with Infiniband as well as Ethernet. Jobs were executed on both interconnects. Results obtained on Infiniband were 1.8 to 2.6 times faster than Ethernet. Average speedup obtained with Infiniband was 2.2X.

Keywords: algorithms; architectures; bioinformatics; BLAST; sequence alignment;

I. INTRODUCTION

Bioinformatics is multidisciplinary fields in which computers are used to store, analyze and manipulate molecular data [1, 2]. As this is the field with very fast growth so a huge amount of data has become available in last two decades [3]. Bioinformatics is a much diversified field with different categories of the task executed on biological data. Some of these tasks are phylogenetic analysis, sequences profiling searching, protein structure prediction, alignment and comparison of biology sequences and so on. Bioinformatics can further be subdivided into two sub disciplines:

- Algorithm development and
- Data analysis and interpretation

Researchers process biological data (e.g., nucleic acid and protein sequences, structures, functions, pathways, and interactions) with the help of various bioinformatics algorithms to identify required information [4]. The fundamental objective is always to reduce analysis and processing time of the algorithms so that different type of decisions can be taken on time. To complete the urge of high performance different computing techniques have been applied to accelerate bioinformatics applications like parallel computing, distributed computing etc. Some applications have also been accelerated using other methods like parallel architectures including Graphical Processing Units (GPUs) which consist of multiple computational units and improve the execution time of bioinformatics algorithms to a very high degree [5]. Distributed Computing is one of the famous application acceleration methods for bioinformatics applications. In distributed processing where the computational process is accelerated by distributing the task among multiple processing units, at the same time various network resources like communication media, network interconnect acts as limiting factors. So rate of acceleration also depends upon the speed of communication media to carry data from one node to

another node, as well as communication, interconnects used to connect the various nodes of a distributed system.

II. PARALLEL BIOINFORMATICS APPLICATIONS AND HARDWARE ARCHITECTURES

In this section parallelized versions of various bioinformatics algorithms, tools along with the architecture implemented on are discussed. The impact of network resources on the performance of parallelized application is also analyzed in each case.

A. mpiBLAST

BLAST is the most commonly used algorithm to identify the similarity between the query sequence and database sequences. When a query sequence is submitted for search, the motive is to find similar sequences from the available databases. Similar sequences are identified by calculating the pair wise score of the alignments between the query sequence and the database sequences, as it is done in the Smith – Waterman algorithm. BLAST uses some heuristic methods to reduce the running time with little sacrifice in accuracy [4].

The mpiBLASTis an accelerated version of BLAST. Execution of this BLAST implementation is divided into two steps. In the first step, the database is divided into various segments and mounted on a shareable storage device. In a second step the queries are executed on different nodes. Different fragments are assigned to each node by following the algorithm with which the number of fragment copies should be least for each search process. mpiBLAST has been benchmarked on several systems to find out its performance and scalability. It is being observed that when database size is larger than the core memory of single node mpiBLAST can achieve a super – linear speed up on multiple nodes. During scaling of the mpiBLAST to many nodes, a decrease in the performance was observed. This performance degradation is caused by five main components: (1) MPI and mpiBLAST initialization, (2) database – fragment copying time, (3) BLAST search time, (4) communication time and (5) result merging time [5, 6].

In mpiBLAST, communication time between the different processing node also one of the performance limiting factors. Communication time among the different nodes can be improved by using faster communication media.

B. mpiBLAST for the Blue Gene/L

In this implementation of BLAST along with database segmentation query distribution was also done to utilize the maximum possible number of nodes of the Blue Gene/L. In Blue Gene/L a large number of low power processors integrated with high speed interconnection networks. So this implementation is very suitable for the highly scalable applications [7].

In Blue Gene implementation of BLAST, as highlighted by the authors, a high speed communication media was used to improve the performance and scalability of the implementation.

C. TurboBLAST

TurboBLAST is a concurrent version of BLAST. It is appropriate to run on heterogeneous clusters of PCs. It is also suitable to be executed on Macintosh computers. With the help of distributed Java "harness" TurboBLAST Divides and Distribute Jobs into small parts, process all parts concurrently and combine the results into the single output file. Following operations are performed on multiple machines:

- BLAST activities are created to search a small set of query sequences against a fragment of the database. These activities are executed in parallel on different nodes of the cluster.
- Standard NCBI BLAST programs complete each activity.
- Results from different activities are compiled into a single output file.

This technique was implemented with the help of a three - tier system consisting of: a Client, a Master and a number of Workers. The Client node fragments the BLAST job into activities to be executed in parallel. These fragments are submitted to the master node. The master node further retrieves the results and compile into one output file. The Master node application and client node applications are written in JAVA [6].

In TurboBLAST a superliner speeds up with respect to the number of workers was achieved when some of the overheads were ignored. These overheads were communication time and job distribution time.

D. BLAST implementation on BEE2

The hardware design of BEE2 system is parallel consisting of processing elements, memory elements, and interconnects. FPGA chips are used as processing elements DRAM modules are used as memory elements. There are two types of interconnects i.e. local and global. In BEE2 design FPGAs are used instead of microprocessors. This implementation also consists of some other components i.e. clock distribution, bootstrap, power regulation and thermal regulation. The performance capability of the BEE2 implementation of BLAST was observed to be 2 times higher in magnitude any of the other systems of the time [8].

In this implementation, two types of connections i.e. local connections and global connections were used. So the speed of these connections duly affects the performance of TurboBLAST at two different levels. To have better performance from this parallel implementation of bioinformatics applications these network connections need to be time efficient.

E. ClustalW-MPI

ClustalW is a tool used for aligning multiple nucleotides or protein sequences. The alignment process is carried out in three different phases, achieved via three steps:

- Pairwise alignment,
- Guide-tree generation
- Progressive alignment.

Parallel and distributed version of ClustalW is ClustalW-MPI, where Message Passing Interface (MPI) is used for communication among distributed processed. The time required for execution is reduced by parallelizing all the three phases of ClustlW. ClustalW-MPI has been designed for the workstation of clusters with distributed memory architecture. The process of distance matrix calculation is parallelized by allocating different parallel for time - independent tasks. Once distance matrix is obtained, neighbor-joining method is used for the generation of guide tree. For the final phase of progressive alignment fine – grained and coarse- grained approaches are used together. As a result, a good increase in performance has been achieved for multiple alignments with low cost PC clusters [9].

In this implementation, distributed memory architecture is being used, so huge volume of data is exchanged among different nodes of the parallel system. A low speed communication media and network interconnects will deteriorate the performance of the application adversely.

F. CUDASAW++2.0

With the help of Smith – Waterman algorithm, the similarity between two sequences is identified by calculating a maximum score for local alignment. If two sequences S1 with length l1 and S2 with length l2 are given, the Smith-Waterman computes the similarity score H(i, j) of two sequences ending at position i and j of S1 and S2, respectively.

CUDA is an extension of C/C++ used for writing multi- threaded code GPUs [10]. A program in CUDA is written in two parts:

- A host program
- A kernel, which is executed by multiple threads of the GPU concurrently [11].

CUDASAW++2.0 is an enhanced version of Smith-Waterman algorithm executed on CUDA enabled GPUs by using Single Instruction Multiple Thread and virtualized Single Instruction Multiple Data abstractions. It is coded in CUDA and C++ programming language. It has achieved a significant performance gain over Smith-Waterman Algorithm by using CUDA - enabled low - cost GPUs [12].

G. MPIPairwiseStatsig

MPIPairwiseStatSig is a parallel tool algorithm used for pair-wise statistical significance Estimation. It is coded in C and MPI. With MPIPairwiseStatSig most compute-intensive segments of the pairwise statistical significance estimation procedure are distributed among multiple processors and a significant speed-ups are achieved for various applications. During the pairwise statistical significance estimation procedure, each shuffling operation like alignment operation is independent. So this procedure is represented as N independent shuffled alignments in this algorithm. [13]

III. CLOUD TECHNOLOGIES

To execute multiple tasks with minimal coupling is one of the most popular trends in most of the bioinformatics applications. Various technologies i.e. schedulers and Map Reduce are used to run these multi-task applications in parallel. Two cloud technologies which are used in many bioinformatics applications are Dryad/DryadLINQ and Apache Hadoop.

A. Dryad/DraydLINQ

Dryad is a distributed engine used for the execution of coarse grain data parallel applications. In this for the task of computation, style of Map Reduce programming is combined with data-flow graphs. It considers any computational problem as a directed acyclic graph (DAG) where different nodes represent computing tasks and edges represent communication between various tasks. The data are stored in (or partitioned to) local disks via the Windows shared directories and metadata files and Dryad schedules the execution of vertices depending on the data locality. The current academic release of DryadLINQ runs only on the Windows HPC Server 2008 operation system [14]

B. Apache Hadoop

In Apache Hadoop data is accessed via HDFS[4], in which all the local disks of the compute nodes are mapped to a single-file system hierarchy, so data is distributed among all data in processing nodes. In HDFS same data is replicated on multiple nodes so that in the case of failure of a particular node system is not to be affected. With this approach fault, handling mechanism is simplified. But one shortcoming of this technology is the communication overhead especially for the applications in which small units of data is produced frequently to be passed between processes running on different nodes. Apache Hadoop runs only on Linux operating systems [14].

IV. BENCHMARKS FOR DISTRIBUTED BIOINFORMATICS APPLICATIONS

As in most of the bioinformatics applications, a huge volume of data is to be dealt with. So execution time is always a critical factor in bioinformatics applications. In all the above distributed bioinformatics applications, considerable speedups are obtained using parallel processing. But in distributed processing communication delay always affected the performance of the applications. Communications delay directly depends upon the characteristics of the communication resources. It depends upon the speed of the communication media. If communication speed is more automatically the delay will be less. So some benchmarks need to be considered for distributed bioinformatics applications as they have an unavoidable affect on the performance. These are mentioned below:

- Speed
- Delay
- Bandwidth

The value of these benchmarks will vary according to the characteristics of the communication interconnects as infiniband is faster than Ethernet. To execute multiple independent jobs or jobs comprising of a large number of tasks with minimal inter-task communication is a very common requirement in most of the bioinformatics applications. So performance also needs to be analyzed for jobs of varying sizes with respect to different communication interconnect for bioinformatics applications.

V. METHODOLOGY

In present work jobs of varying size were processed with a distributed bioinformatics application and execution time was analyzed for two different communications interconnects .i.e. Infiniband and Ethernet. The application used to process the jobs is Basic Local Alignment Search Tool (BLAST). BLAST was implemented on a cluster of 10 compute node. Jobs were processed on a database of 1.2 GB. Database segmentation technique was used to implement BLAST. Different nodes of the cluster were connected with Ethernet as well as Infiniband. Red Hat Enterprise Linux 5.6 operating system was installed.

VI. RESULTS AND DISCUSSION

Experiments were performed with jobs (query sequences) of varying size from 500 to 4000 in length to collect the execution time. Results were collected with both interconnects i.e. Infiniband and Ethernet. The execution time of different sized jobs on a cluster of 10 compute nodes with both interconnects is shown in Table 1.

Table 1: Execution time for jobs of varying size on Infiniband and Ethernet

Job Size/ Query Length (Characters)	Execution Time(Sec)	
	Infiniband	**Ethernet**
500	458.3	816.4
1000	897.1	1923.4
1500	1453.6	3233.2
2000	1823.4	4355.3
2500	2324.1	5384.4
3000	2902.3	7498.7
3500	3633.2	7987.2
4000	4145.3	8734.1

As shown in Table 1, as the job size increases, processing time also increases. Execution time for jobs of different sizes on Infiniband connected cluster is less than Ethernet connected cluster. These results are depicted with the help of a bar graph in Figure 1.

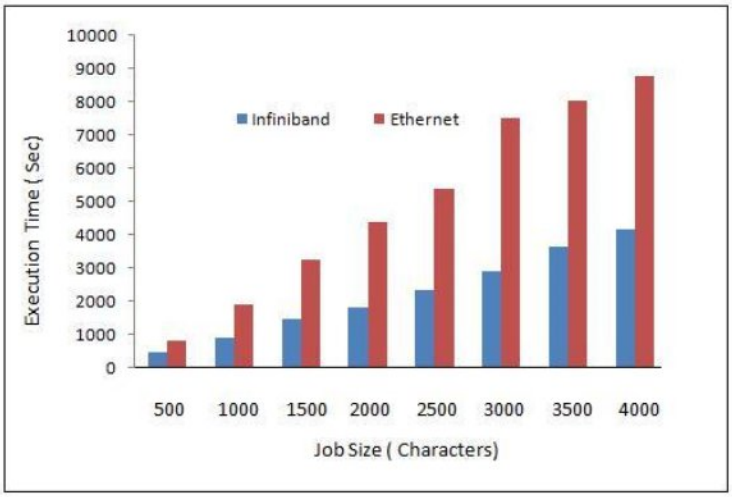

Figure 1: Execution time for jobs of varying size on Infiniband and Ethernet connected cluster (Bar Graph).

It can be easily interpreted from Figure 1 that computing cluster connected by Infiniband is faster than Ethernet connected cluster for bioinformatics applications.

Speed-up obtained for jobs of different sizes on Infiniband connected cluster in-comparison to Ethernet connected cluster are shown in Table 2.

Table 2: Speed-up obtained with Infiniband in comparison to Ethernet

Job Size/ Query Length (Characters)	**Speed-up**
500	1.8
1000	2.1
1500	2.2
2000	2.4
2500	2.3
3000	2.6
3500	2.2
4000	2.1

For jobs of different sizes on a database of 1.2 GB, Infiniband cluster was 1.8 to 2.6 times faster than Ethernet cluster. An average speed-up obtained with Infiniband cluster was 2.2 times in contrast to Ethernet cluster. Data shown in Table 2 is depicted with a line graph in Figure 2.

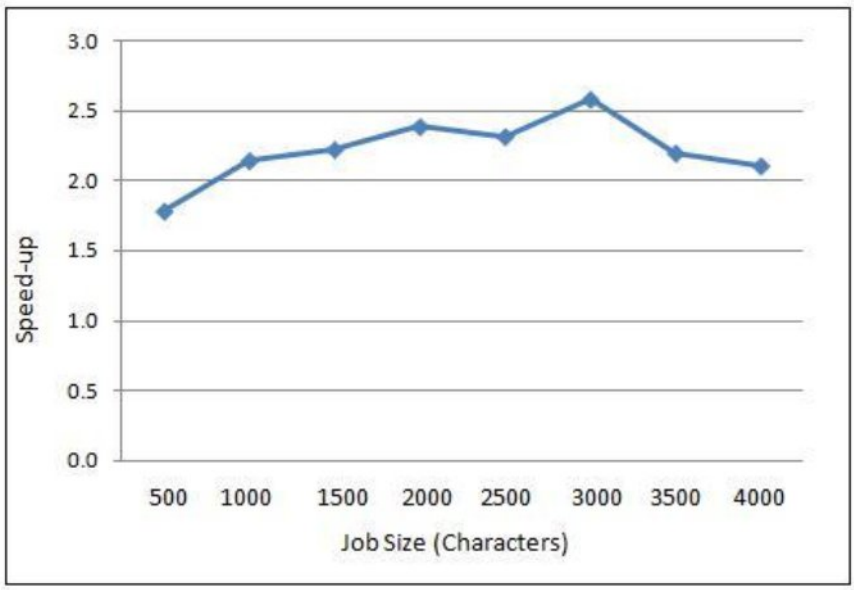

Figure 2: Speed-up obtained with Infiniband Cluster in Comparison to Ethernet Cluster (Line Graph)

CONCLUSIONS

By studying various parallelized implementations of bioinformatics applications, it is being analyzed that these applications are highly improved. A considerable speedup is being obtained in almost each implementation. But in each of the parallelized application, performance is highly affected by the speed of network interconnects and communication media. As we have different parallel architectures i.e. distributed memory and shared memory, which require a varying quantity of data to be exchanged among processing nodes. Distributed memory implementations have more adverse effects as more data is transferred among the processing nodes. In our work, it is being observed that parallel systems connected with Infiniband are more suitable for bioinformatics applications as compare to Ethernet. When BLAST application processed on Infiniband connected cluster on an average it gave 2.2 times faster results as compare to Ethernet backbone.

ACKNOWLEDGEMENTS

Authors express deep gratitude to the Dean, Research, Innovation and Consultancy Deptt. of I.K.G. Punjab Technical University, Kapurthala for giving the opportunity to carry on this research work.

REFERENCES

[1] B. K. Pandey, S.K. Pandey and D. Pandey, " A Survey of Bioinformatics Applications on Parallel Architectures", International Journal of Computer Applications , vol. 23, no. 4 , pp. 21-25, June, 2011.

[2] N.M. Luscombe, D. Greenbaum and M. Gerstein ," What is Bioinformatics? A proposed definition and Overview of the Field", Method Information Med, pp. 346-358, April, 2001.

[3] S.K. Pal and S.S. Ray, " Evolutionary Computation in Bioinformatics: A Review", IEEE trans. On Systems, Man and Cybernetics, vol.36, no. 5, pp. 601-615, September, 2006.

[4] O. Trelles, " On the parallelization of Bioinformatics Applications" , Journal of Molecular Biology, vol. 12, no. 4, pp. 27-42, 2006.

[5] J. Varre, B. Schmidt, S. Janot and M. Giraud , " Manycore high-performance computing in Bioinformatics", Advances in Genome Sequence Analysis and Patteren Discovery, chapter 8, 2011.

[6] R.D. Bjomson, A.H. Sherman, S.B. Weston, N. Willard and J.Wing, " TurboBlast : A Parallel Implementation of BLAST Built on the TurboHub", IEEE Distributed and Parallel Processing Symposium, pp. 183-190, 2002.

[7] H. Rangwala, E. Lantz, R. Musslemen, K. Pinnow and B. Smith, " Massively Parallel BLAST for the Blue Gene/L", High Availability and Performance Computing Workshop, pp. 1-6, 2005.

[8] C. Chang, "BLAST Implementation on BEE2", IEEE Symposium on Workload Characterization" pp. 73-79, 2003.

[9] K. Li, "ClustalW-MPI: ClustalW Analysis using Distributed and Parallel Computing", Bioinformatics, vol. 19, no. 12, pp. 1585-1586, March, 2003.

[10] J. Nickolls, I. Buck, M. Garland and K. Skadron, "Scalable Parallel Programming with CUDA", ACM Queue, vol. 6, no.2, pp. 40-53, 2008.

[11] E. Lindholm, J. Nickolls, S. Oberman and J. Montrym, "NIVIDIA Tesla: A unified graphics and computing architecture", IEEE Micro, vol. 28, no. 2, pp. 39-55, 2008.

[12] Y. Liu, B. Schmidt and D.L. Maskell, " CUDASW++2.0: enhanced Smith-Materman protein database search on CUDA- enabled GPUs based on SIMT and virtualized SIMD abstractions", BMC Research Notes, vol. 3, pp. 1-12, 2010.

[13] A. Agrawal, S. Misra, D. Honbo and A. Choudhary, " MPIPairwiseStatSig: Parallel Pairwise Statistical Estimation of Local Sequence Alignment", High Performance Data Computing, pp. 470-475, 2010.

[14] J. Ekanayake and J. Qiu, "Cloud Technologies for Bioinformatics Applications", IEEE Transaction on Parallel and Distributed Systems, vol. 22, no. 6, pp. 998-1011, June, 2011.